Naïve Bayes Machine Learning Classification with R Programming: A case study of binary data sets

Yagyanath Rimal[1]

[1]*School of Engineering, Pokhara University, Nepal*

E-mail: rimal.yagya@pu.edu.np

---***---

Abstract - *This analytical review paper clearly explains Naïve Bayes machine learning techniques for simple probabilistic classification based on bayes theorem with the assumption of independence between the characteristics using r programming. Although there is large gap between which algorithm is suitable for data analysis when there was large categorical variable to be predict the value in research data. The model is trained in the training data set to make predictions on the test data sets for the implementation of the Naïve Bayes classification. The uniqueness of the technique is that gets new information and tries to make a better forecast by considering the new evidence when the input variable is of largely categorical in nature that is quite similar to how our human mind works while selecting proper judgement from various alternative of choices and can be applied in the neuronal network of the human brain does using r programming. Here researcher takes binary.csv data sets of 400 observations of 4 dependent attributes of educational data sets. Admit is dependent variable of gre, score gpa and rank of previous grade which ultimately determine whether student will be admitted or not for next program. Initially the gra and gpa variables has 0.36 percent significant in the association with rank categorical variable. The box plot and density plot demonstrate the data overlap between admitted and not admitted data sets. The naïve Bayes classification model classify the large data with 0.68 percent for not admitted where as 0.31 percent were admitted. The confusion matrix, and the prediction were calculated with 0.68 percent accuracy when 95 percent confidence interval. Similarly, the training accuracy is increased from 29 percent to 32 percent when naïve Bayes algorithm method as use kernel is equal to TRUE that ultimately decrease misclassification errors in the binary data sets.*

Key Words: *Naive Bayes Classifier, Supervised Learning.*

1. INTRODUCTION

Machine learning is the study of algorithms that can learn from input data and therefore predict data for future analysis. They use a complex model and an algorithm they learn by themselves to predict, which in commercial use is known as predictive analysis [1]. Based on the statistical data of the statisticians in the form of variables, the data were processed through traditional programming [2]. Google, Facebook and other social networks used to a large extent the classification algorithm of automatic design pages. Although there were two types of supervised learning and an unsupervised algorithm [3]. The supervised learning mainly focuses on the classification of the set of identification categories that support the set of training data that contain observations that, at a certain point, classify the variables of the research data into different classes. It is a classification technique based on Bayes' theorem with a hypothesis of independence among predictors [4]. In simple terms, a Naive Bayes classifier assumes that the presence of a particular attribute in a class is not related to the presence of any other implicit characteristic that the evidence is based on evidence [5]. The naive Bayes classifier has its fundamental pillar from the concept of Bayes' theorem explained by probability theory.

The probability is the possibility that an event of the various cases occurs. The probability can be related to our normal life and helps us solve many real problems. There is a probability that a single event is calculated as the proportion of cases in which that particular event occurs. Similarly, we have a probability of a group of events, calculated as the proportion of cases in which the group of events occurs together [6]. In the same way, the individual probability was also calculated, so all the yes and the probabilities were calculated first, then all the yes forecasts and not multiplied, so these records were divided with a normalized set of yes data and a total of n. Finally, consider the probability that yes and no, if yes, it is greater than no case, then yes, otherwise there will be no forecasting. For the simplest example, the football players' dataset can play in different seasons like windy, moderate, summer, winter, summer, monsoon, like many categories where there were many conditions of probability of game habits that can having many conditions, there could also be a combination of several cases When calculating the probability table, the entire frequency was calculated in each case and predicted its probability [3].

	play	
	Yes	No
Summer	3	2
Monsoon	4	0
Winter	2	3

Likelihood		play			
		Yes	No	∑/Tot	P(x|c) Summer yes 3/9=0.33
Season	Summer	3/9	2/5	5/14	P(x) Summer 5/14=0.36
	Monsoon	4/9	0/5	4/14	
	Winter	2/9	3/5	5/14	
		9/14			
		P(c) yes 9/14=0.64			

Therefore, the probability of occurring in the summer as P (c | x) = P (Yes) | Summer = P (Summer | Yes) * P (Yes) /P (Summer) = 0.33 * 0.64 / 0.36 = 0.62.i.e (2/9) * (6/9) *(9/14) / (5/14) * (7/14) * (8/14) = 0.622 which is always compared with the threshold probability test of 0.5 for the forecast. Therefore, the naive Bayes classification tries to predict the situation in many alternative cases as the wind itself, but not the sunny cases therefore: P (C | X) = P (x | c) * P (c) / P (x) the product of probability and the previous probability of the class divided by the previous probability of the predictor. Here, P (c) is a previous probability that describes the degree to which we believe that the accuracy of the model describes reality based on all our previous information. Similarly, p (x) is a normalization constant that causes the subsequent density to be integrated into one. The posterior probability P (C | X) which represents the degree to which we believe that a given model accurately describes the situation, given the available data and all our previous information. Similarly, P (x | c) is the probability that describes how well the model predicts data [7]. Machine learning is a category of algorithms that allows software applications to be more precise in predicting results without being explicitly programmed. The basic premise of machine learning is to create algorithms that can receive input data and use statistical analysis to predict an output, while outputs are updated as soon as new data become available. The Naive Bayes classification is an important tool related to the analysis of large data or work in the field of data science [8]. R is a free software environment for statistical and graphical processing and is widely used by the academic and industrial world. The Naive Bayes classifier is a simple classifier based on the famous Bayes theorem. Despite its simplicity, it has remained a popular choice for text classification. The Naive Bayes classifier applies the well-known Bayes theorem for conditional probability. In the simplest form for the event AA and BB, the Bayes theorem relates two conditional probabilities in the following way: P (A∩B) = P (A, B) = P (A) P (B | A) = P (B) P (A | B) P (A∩B) = P (A, B) = P (A) P (B | A) = P (B) P (A | B)⟹P (B | A) = P (B) P (A | B) P (A) ⟹P (B | A) = P (B) P (A | B) P (A)

In a classification problem, we have some predictors (characteristics / covariates / independent variables) and a result (target / class / dependent variable) [9]. Each observation has some values for the class of predictors. From these predictors and associated classes, we want to learn so that, if feature values are provided, we can predict the class. Now in Naive Bayes, the algorithm evaluates a probability for each class, when the predictor values are automatically assigned, we can go to the class, which is more likely. The previous expression is quite long and also including several conditional probabilities. But with the assumption of conditional independence, this long expression can be reduced to a very simple form [10]. The assumption of conditional independence is that, given a class, the predictor / characteristic values are independent of each other. There is no correlation between the characteristics of a certain class [11] at any time. Mathematically, if the event AA and BB are independent conditioned by the CC event, then the following is true: P (A | B, C) = P (A | C) P (A | B, C) = P (A | C). Now we are trying to get the probability for certain functionality values. The denominator above is a constant term for certain characteristics. Therefore, it will be sufficient to consider only the part of the numerator to compare the probabilities of different classes conditioned in values of fixed characteristics. That is, we evaluate the part of the numerator for every possible union value and in the end, we vote for the one with the highest value. we need the conditional probability distributions, f (Xj | Ck) f (Xj | Ck) for j = 1,2, ..., nj = 1,2, ..., n. If XjXj is continuous, normality is a common hypothesis. If XkXk is discrete, a common hypothesis is the monomial distribution. Furthermore, we can apply non-parametric distributions. Naive Bayes algorithms are mainly used in the analysis of feelings, spam filters, recommendation systems, etc. They are fast and easy to implement, but their biggest drawback is that predictors must be independent. In most real cases, predictors are dependent, which hampers the performance of the classifier [11].

2. USING R PROGRAMMING

For demonstration purpose, we will make a Niave Bayes classifier here. We will use a data set that contains information of 400 student's database of four attributes with score. Their scores in different subjects and their educational choices (general, academic or vocational). There are other variables indicating their socio-economic status and their gender. We will make a Naive Bayer classifier here.

```
> install.packages("naivebayes")
> library(naivebayes)
> library(naivebayes)
> library(dplyr)
> library(ggplot2)
> library(psych)
> library(caret)
> library(e1071)
> mydata <- read.csv("https://stats.idre.ucla.edu/stat/data/binary.csv")
```

This command used to load data from internet sources with all rows of the data

```
> head(mydata)
  admit gre  gpa rank
1     0 380 3.61    3
2     1 660 3.67    3
3     1 800 4.00    1
4     1 640 3.19    4
5     0 520 2.93    4
6     1 760 3.00    2
> data=mydata
> str(data)
'data.frame':           400 obs. of  4 variables:
 $ admit: int  0 1 1 1 0 1 1 0 1 0 ...
 $ gre  : int  380 660 800 640 520 760 560 400 5
 $ gpa  : num  3.61 3.67 4 3.19 2.93 3 2.98 3.08 3
 $ rank : int  3 3 1 4 4 2 1 2 3 2 ...
> colnames(data)
[1] "admit" "gre"   "gpa"   "rank"
```

Those commands display the first head and its attributes and columns name of data sets of binary .csv

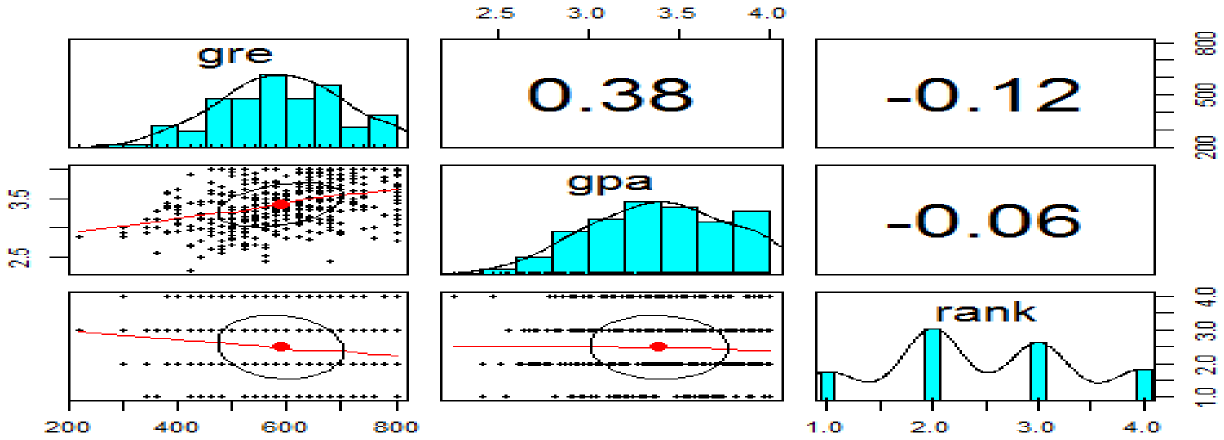
```
> xtabs(~admit+rank,data=data)
     rank
admit     1   2   3   4
          0 28 97 93 55
          1 33 54 28 12
```

Multiway categorical variables there were 28 students who were rank 1 do not admitted in the program similarly there were 33 candidate of same category rank were admitted.

```
> data$rank=as.factor(data$rank)
> data$admit=as.factor(data$admit)
```

The integer type of admit and rank variable were changed into numeric categorical variable.

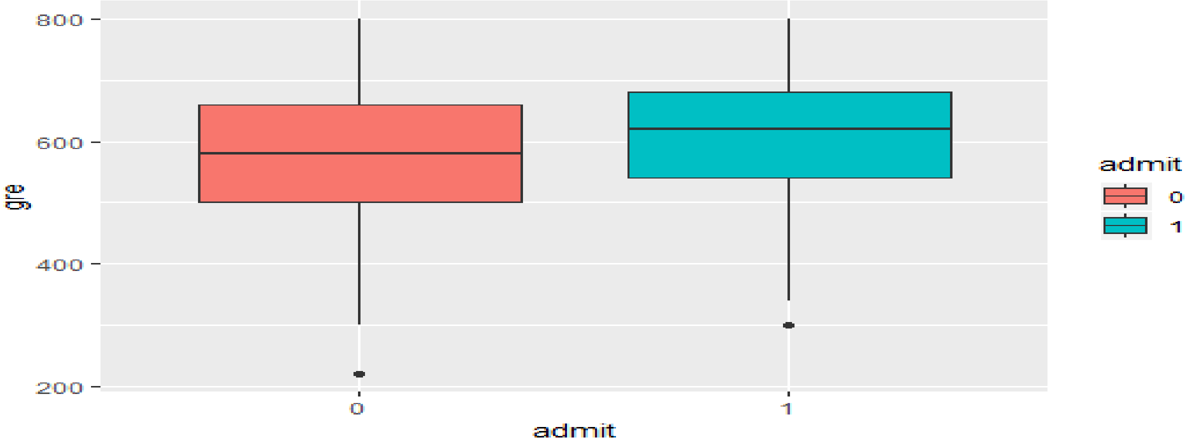
```
> pairs.panels(data[-1])
```



The cross-matrix plot demonstrates the correlation coffined between independent variables where gre and gpa has 0.38 percent association whereas rank has negative association to gpa and gre.

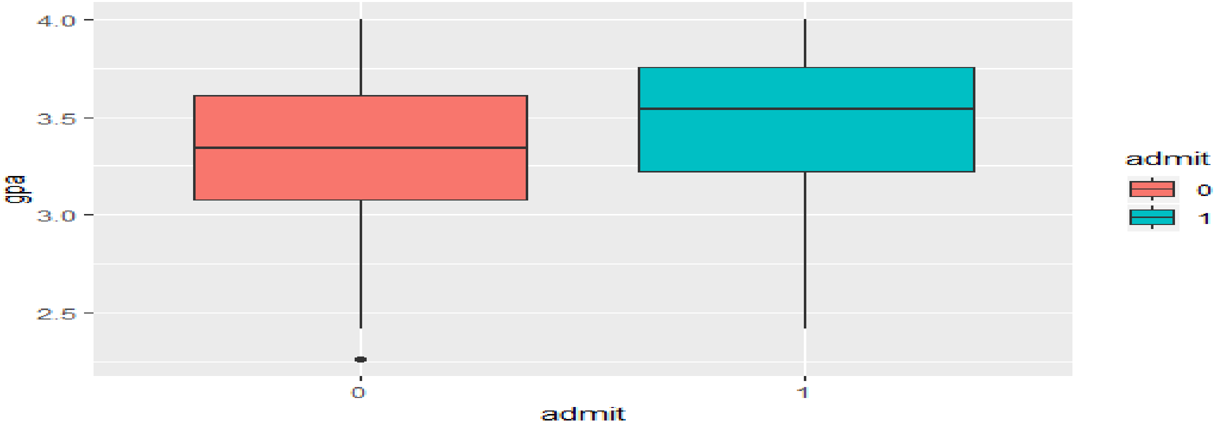
```
> data%>%
+ ggplot(aes(x=admit,y=gre,fill=admit))+ geom_boxplot()+ ggtitle("Box Plot")
```

Fig: 2 Box Plot



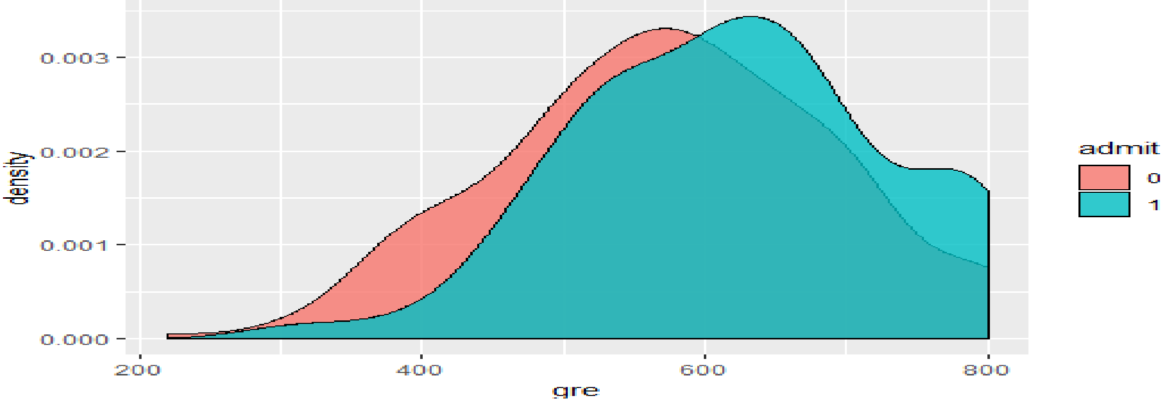
The categorical variable admit is in x axis and y variable gre box plot demonstrate there were not equal distribution of data in each category.

```
> data%>%
+   ggplot(aes(x=admit,y=gpa,fill=admit))+   geom_boxplot()+   ggtitle("Box Plot")
```



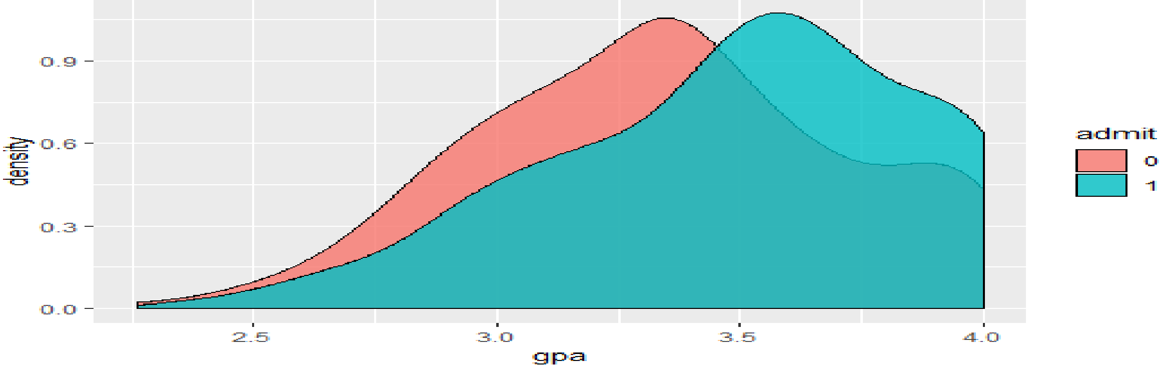
The categorical variable admits in x axis and gpa in y axis plot demonstrate box plot whose data structure who's largely varied in nature.

```
> data%>%
+   ggplot(aes(x=gre,fill=admit))+  geom_density(alpha=0.8,color='black')+   ggtitle("Density Plot")
```



The density plot when overlap in between gre score and admit demonstrate slightly overlap.

```
> data%>%
+   ggplot(aes(x=gpa,fill=admit))+ geom_density(alpha=0.8,color='black')+   ggtitle("Density Plot")
```



The density plot between gpa and admit demonstrates and overlap between the chosen predictors for each category of the target.

```
> set.seed(1234)
> ind=sample(2,nrow(data),replace=T, prob=c(.8,.2))
> train=data[ind==1,]
> test=data[ind==2,]
```

Naive Bayes classifier for binary data sets were grouped into 80/20 partitioning for the training and validation test set partition from caret package to make a balanced partitioning randomly from 400 data observation. This is very important in this context because we are going to calculate the prior probabilities from the count of the training data. So, it should follow the proportion of the parent dataset.

The naïve bayers classification will be created with admit categorical variable with all other variables stored in model.

```
> model=naiveBayes(admit~.,data=train )
> model
Naive Bayes Classifier for Discrete Predictors
Call:
naiveBayes.default(x = X, y = Y, laplace = laplace)
A-priori probabilities:
Y
       0         1
0.6861538 0.3138462
Conditional probabilities:
        gre
Y      [,1]    [,2]
  0 578.6547 116.325
  1 622.9412 110.924
        gpa
Y      [,1]      [,2]
  0 3.355247 0.3714542
  1 3.533627 0.3457057
            rank
Y           1          2          3          4
  0 0.10313901 0.36771300 0.33183857 0.19730
  1 0.24509804 0.42156863 0.24509804 0.08823
```

The model stored 69 percent of not admitted and 31 percent admitted, the gre, gpa and rank were further calculated with its respective percentage.

```
> summary(model)
        Length Class  Mode
apriori   2      table  numeric
tables    3      -none- list
levels    2      -none- character
isnumeric 3      -none- logical
call      4      -none- call
> # normal distribution when independent variable is categorical p(rank1/admit=0)=0.103 and p(rank1/admit=1)=0.245 from the statistic table were taken.
> p=predict(model,test)
```

The naiveBayes function assumed gaussian distributions for numeric variables. Also, the priori is calculated from the proportion of the training data. The priories are shown when the object is printed. The Y values are the means and standard deviations of the predictors within each class. The good thing about this package that we can use Kernel based density for the continuous predictors.

```
> confusionMatrix(table(p,test$admit))
Confusion Matrix and Statistics
p    0  1
  0 47 21
  1  3  4
 Accuracy : 0.68
95% CI : (0.5622, 0.7831)
No Information Rate : 0.6667
P-Value [Acc > NIR] : 0.4566734
Kappa : 0.122
Mcnemar's Test P-Value : 0.0005202
Sensitivity : 0.9400
```

```
Specificity : 0.1600
Pos Pred Value : 0.6912
Neg Pred Value : 0.5714
Prevalence : 0.6667
Detection Rate : 0.6267
Detection Prevalence : 0.9067
Balanced Accuracy : 0.5500
'Positive' Class : 0
```

The prediction with test data is calculated and confusion matrix were calculated and found 68 percentage accuracy result when confidence interval of 95 percentage. Whose p value is less than .05 implied that there is significant rejection the hypothesis. Similarly, the sensitivity is higher than specificity demonstrate data has good enough in model test is not that bad, given that we used just two variables.

```
> train%>%
+   filter(admit=="0")%>%
+   summarise(mean(gre),sd(gre))
  mean(gre) sd(gre)
578.6547 116.325
```

The train model is filtered with admit value is zero no result and mean and standard deviation were calculated.

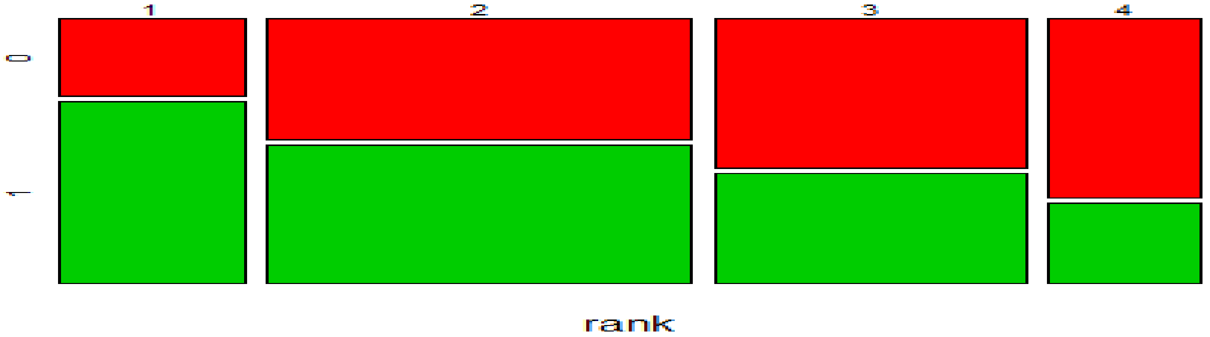
```
> train%>%
+   filter(admit=="1")%>%
+   summarise(mean(gre),sd(gre))
  mean(gre) sd(gre)
622.9412 110.924
```

Similarly, train model is filtered with admit value is zero no result and mean and standard deviation were calculated.

```
> model2=naive_bayes(admit~.,data=train, usekernel = T)
```

Another way of calculating naïve bayers classification demonstrate that it raised the training accuracy a little higher, with no change in the test set

```
> plot(model2)
```



This plot demonstrates that the green color graphs were admitted and red color were not admitted in each rank category.

```
> p1=predict(model,train)
> head(cbind(p1,train))
  p1 admit gre  gpa rank
1  0     0 380 3.61    3
2  0     1 660 3.67    3
3  1     1 800 4.00    1
4  0     1 640 3.19    4
6  0     1 760 3.00    2
7  0     1 560 2.98    1
```

The prediction and cbind combines model and prediction variables.

```
> (tab1=table(p1,train$admit))
```

```
p1      0   1
   0    196  69
   1    27  33
```

This confusion matrix demonstrates the main diagonal elements were valid in machine prediction as well as model prediction whereas second diagonal elements were misclassification needs more correction on data sets.

```
> 1-sum(diag(tab1))/sum(tab1)
[1] 0.2953846
> p2=predict(model,test)
> (tab2=table(p2,test$admit))
p2   0  1
   0 47 21
   1  3  4
> 1-sum(diag(tab2))/sum(tab2)
[1] 0.32
```

There were 29 and 32 percent errors in train and test data sets were easily calculated, therefore naïve probability model fitting is best when the data sets were in categorical numeric type using machine learning process.

3. CONCLUSION

A Naive Bayes classifier is a probabilistic machine learning model used for the classification activity. The root of the classifier is based on the Bayes theorem. The Bayesian algorithm provides a probabilistic framework for a classification problem. It has a simple and solid base for modeling data and is quite robust with outliers and missing values. However, this algorithm is widely implemented in text mining and document classification where the application has a large set of attributes and attribute values to calculate. It also serves as a good reference point for comparison with other models. The implementation of the Bayesian model in production systems is quite simple and the use of data mining tools is optional. An important limitation of the model is the assumption of independent attributes, which can be mitigated by advanced modeling or by decreasing the dependency between attributes by pre-processing.

REFERENCES

[1] Dodge, M. (2008). Understanding Cyberspace Cartographies:. A thesis submitted for the degree of Doctor of Philosophy.

[2] D. Liske, "Lyric Analysis: Predictive Analytics using Machine Learning with R," 2018.

[3] A. Sharma, "How Different are Conventional Programming and Machine Learning?," Tags: Machine Learning, Programming, 2018.

[4] J. Brownlee, "Supervised and Unsupervised Machine Learning Algorithms," 2016.

[5] M. Sidana, "Types of classification algorithms in Machine Learning," 2017

[6] M. D. B. J. M. D. L. R. C. P. C. A. S. B. B. Troy CS, "Sequence - Evolution - Function: Computational Approaches in Comparative Genomics.," 2001.

[7] J. P. Jiawei Han, "Learn more about Naïve Bayesian Classifier," 2012.

[8] L. S. R. Arboretti Giancristofaro, MODEL PERFORMANCE ANALYSIS AND MODEL VALIDATION, 2003.

[9] S. Arora, "Data Science vs. Big Data vs. Data Analytics," 2015.

[10] B. Deshpande, "2 main differences between classification and regression trees," 2011.

[11] R. Khan, "Naive Bayes Classifier: theory and R example," 2017.

[12] M. D. S. D. K. Malik Yousef, Feature Selection Has a Large Impact on One-Class Classification Accuracy for MicroRNAs in Plants, 2016.